Multilevel optimization : application to shape optimum design with a one-shot method

Nathalie Marco, François Beux

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Multilevel Optimization : Application to One-Shot Shape Optimum Design

Nathalie MARCO and François BEUX

N° 2068

Octobre 1993

PROGRAMME 6

Calcul scientifique,
modélisation
et logiciels numériques

ISSN 0249-6399

Multilevel Optimization : Application to One-Shot Shape Optimum Design

Nathalie MARCO *and François BEUX **

Programme 6 — Calcul scientifique, modélisation et logiciel numérique
Projet Sinus

Rapport de recherche n ° 2068 — Octobre 1993 — 30 pages

Abstract: Gradient method is applied to the optimal control of a system for which each simulation is expensive. Instead of solving completely the flow equation as a state equation in the shape optimization, we use a "one-shot method" which solves simultaneously the system optimality. It is tested for the problem of shape optimization of a nozzle in a 2D Euler flow.

(Résumé : tsvp)

*INRIA Sophia Antipolis,B.P. 93, 06902 Sophia-Antipolis Cedex,France
**Dipartimento di Ingegneria Aerospaziale,Università degli Studi di Pisa,V. Diotisalvi, 2, 56100 Pisa, Italia

Unité de recherche INRIA Sophia-Antipolis
2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex (France)
Téléphone : (33) 93 65 77 77 – Télécopie : (33) 93 65 77 65

Optimisation Multi-niveau : Application à l'optimisation de formes par une méthode "One-Shot"

Résumé : Nous considérons la méthode de gradient appliquée au contrôle optimal d'un système pour lequel chaque simulation est coûteuse. Au lieu de résoudre complètement l'écoulement comme une équation d'état dans un problème d'optimisation de forme, nous effectuons une résolution simultanée ("méthode one-shot") des équations d'état et d'état-adjoint et de la condition d'optimalité. La méthode est testée sur un problème d'optimisation de forme d'une tuyère plongée dans un écoulement 2D régi par les équations d'Euler .

Contents

Introduction

The research of an optimal shape in aerodynamics is a problem which has been attracting for a long time physicists and mathematicians. The development of computers and the progress achieved in numerical calculation have brought a new interest for these problems and have allowed to obtain numerical experiments for models more and more complex; development of methods allowing this sort of calculation is the subject of Optimum Design in aerodynamics.

Many experiments have been performed with models with increasing complexity, up to full potential flows (see [8] and [11]). The main difficulty is related to the already large cost of one flow calculation. Moreover, this cost must be multiplied by the points of design (several Mach regimes,...) and the number of cost evaluations performed for optimization. Conversely, choosing a simpler model is a delicate task if we want to be sure that the apparent improvement of the shape is not invalidated by the coarseness of the model.

In many simplified cases, the optimization is reduced to an inverse problem, that can, ideally, be just as costly as a single simulation; however, in the most general cases, it seems necessary to organize the optimization as a succession of simulations with increasingly better parameters deduced by an optimization software kernel. In that general case, the simulation code is considered as a black box. The most reluctant consequence is that finding N optimal parameters will be paid at least by N simulations.

For aerodynamics, 2D full potential flows begin to be used in [1], [7], [11], [13]; often less than 10 parameters are used.

If we examine more accurately the CPU cost of such an optimization, we note that a cost which is a linear function of the number N of unknown shape parameters, is an ideal situation that generally is not attained. By contrary, the problem stiffness increases with the number N and is paid by a N^α complexity ($\alpha > 1$).

In this paper, we consider the optimal design of a nozzle occupied by a 2-D subsonic Euler flow, as in [4] and [5]. Defining the optimal shape of an object as the minimum of a "cost" functional, we try to approach it as close as possible by using gradient methods. In [2], [4], [5], the **State Equation**

and the **Optimality Condition** have been exhibited. Now, we introduce a new point of view: instead of solving completely each complex linear system which demands a lot of calculations of the cost functional (that are in fact not really necessary), we **solve simultaneously** the linear systems (this is called a one-shot method, after [14]). In [14] and [15] this method has already been introduced by Sh. Ta'asan who presents efficient results.

After having introduced Euler equations in the first section, the continuous and discrete problems in section 2, we present in sections 3 and 4 the one-shot method and its performances.

1 Euler Equations

The general form of the steady Euler system is :

$$F(W)_x + G(W)_y = 0 \tag{1}$$

It can also be written as :

$$A(W)W_x + B(W)W_y = 0 \tag{2}$$

where :

$$A(W) = \frac{dF}{dW}(W) \quad \text{et} \quad B(W) = \frac{dG}{dW}(W)$$

The components of W, which is a vector field defined in $\mathcal{C}^2(I\!R^2, I\!R^4)$, are the following conservative variables:

$$W = (\rho, \rho u, \rho v, E)$$

and $F(W)$ and $G(W)$ are the flow functions :

$$F(W) = \begin{pmatrix} \rho\ u \\ \rho\ u^2 + P \\ \rho\ u\ v \\ (E+P)u \end{pmatrix} \qquad G(W) = \begin{pmatrix} \rho\ v \\ \rho\ u\ v \\ \rho\ v^2 + P \\ (E+P)v \end{pmatrix}$$

The pressure of the Euler flow is :

$$P(W) = (\frac{c_p}{c_v} - 1)(E - \frac{1}{2}\ \rho\ (u^2 + v^2))$$

where ρ is the density, u and v are the horizontal and vertical velocity components and E is the total energy per volume unit.

2 The optimization problem

2.1 Boundary conditions and variational formulation

A slip condition is imposed on the lower and upper sides : $\vec{U}.\vec{n} = 0$. At the entry and the exit, are imposed Dirichlet conditions.

Let Γ_{ad} be a set of parametrization included in $\mathcal{C}^2([0, 2\Pi], I\!R^2)$; at any γ in Γ_{ad}, corresponds a regular subdomain Ω_γ of $I\!R^2$, whose boundary is also denoted by γ (see Figure 1).

Variational formulation of the Euler Equations (1) is :

$$-\int\int_{\Omega_\gamma} (F\,\varphi_x + G\,\varphi_y)\,dxdy + \int_\gamma \begin{pmatrix} 0 \\ P\,n_x \\ P\,n_y \\ 0 \end{pmatrix} \varphi\,d\sigma = 0 \quad \forall\,\varphi \in D(\Omega_\gamma) \quad (3)$$

$$\Leftrightarrow \quad (\Psi(W)\ ,\ \varphi) = 0 \quad \forall\,\varphi \in D(\Omega_\gamma) \quad (4)$$

where (. , .) represents the duality between $D'(\Omega_\gamma)$ and $D(\Omega_\gamma)$, $\Psi(W)$, the Euler flow, belongs to) $D'(\Omega_\gamma)$ and φ is a test function.

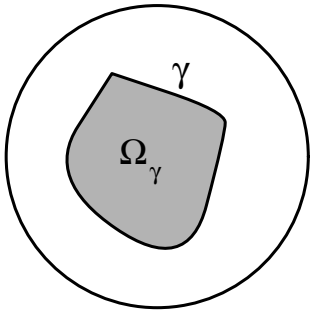


Figure 1: Continuous problem subdomain

2.2 Continuous problem

By considering the following functional :

$$\begin{aligned} J : \Gamma_{ad} \times \mathcal{C}^2(I\!R^2, I\!R^4) &\longrightarrow I\!R \\ (\gamma, W) &\longmapsto J(\gamma, W(\gamma)) \end{aligned}$$

where W is the state, solution of (4) :

$$\begin{array}{rcl} W : \Gamma_{ad} & \longrightarrow & \mathcal{C}^2(I\!R^2, I\!R^4) \\ \gamma & \longmapsto & W(\gamma) \end{array}$$

and

$$J(\gamma, W(\gamma)) = \int_\gamma (P(W(\gamma)) - P^d)^2 d\sigma$$

with P^d the desired pressure.

Finally, the "cost" functional is defined as :

$$\begin{array}{rcl} j : \Gamma_{ad} & \longrightarrow & I\!R \\ \gamma & \longmapsto & j(\gamma) = J(\gamma, W(\gamma)) \end{array}$$

The problem is to minimize j :
we are looking for the control variable $\gamma_o \in \Gamma_{ad}$ such that :

$$j(\gamma_o) = \min_\gamma j(\gamma)$$

Gradient of the cost functional j :

Let $\delta\gamma$ be an element of Γ_{ad}. Then, the Fréchet-derivative of j is :

$$\frac{dj}{d\gamma}(\gamma)\delta\gamma = \frac{\delta J}{\delta\gamma}(\gamma, W)\delta\gamma \; + \; < \; \frac{\delta J}{\delta W}(\gamma, W), \frac{dW}{d\gamma}(\gamma)\delta\gamma \; >$$

where $<\ ,\ >$ represents the duality between $(\mathcal{C}^2(I\!R^2, I\!R^4))'$ and $\mathcal{C}^2(I\!R^2, I\!R^4)$. Using the differentiation of $\Psi(\gamma, W(\gamma))$, we get :

$$\frac{dj}{d\gamma}(\gamma)\delta\gamma = \frac{\delta J}{\delta\gamma}(\gamma, W)\delta\gamma \; - \; < \; \frac{\delta J}{\delta W}(\gamma, W), (\frac{\delta \Psi}{\delta W}(\gamma, W))^{-1} \; \frac{\delta \Psi}{\delta\gamma}(\gamma, W)\delta\gamma \; >$$

or :

$$\frac{dj}{d\gamma}(\gamma)\delta\gamma = \frac{\delta J}{\delta\gamma}(\gamma, W)\delta\gamma \; - \; < \; ((\frac{\delta \Psi}{\delta W}(\gamma, W))^{-1})^* \frac{\delta J}{\delta W}(\gamma, W), \frac{\delta \Psi}{\delta\gamma}(\gamma, W)\delta\gamma \; >$$

where the star holds for the adjoint operator. So in conclusion :

$$\frac{dj}{d\gamma}(\gamma)\delta\gamma = \frac{\delta J}{\delta\gamma}(\gamma, W)\delta\gamma \; - \; < \; \Pi, \frac{\delta \Psi}{\delta\gamma}(\gamma, W)\delta\gamma \; > \qquad (5)$$

Π is the **adjoint state** solution of :

$$(\frac{\delta \Psi}{\delta W}(\gamma, W))^* \Pi = \frac{\delta J}{\delta W}(\gamma, W) \tag{6}$$

In the Optimization Problem, we have to solve :

$$\left\{ \begin{array}{rcl} \Psi(\gamma, W(\gamma)) & = & 0 \\ (\frac{\delta \Psi}{\delta W}(\gamma, W(\gamma)))^* \Pi(\gamma) & = & \frac{\delta J}{\delta W}(\gamma, W(\gamma)) \\ j^{'}(\gamma) & = & \frac{\delta J}{\delta \gamma}(\gamma, W(\gamma)) - < \Pi(\gamma), \frac{\delta \Psi}{\delta \gamma}(\gamma, W(\gamma)) > \end{array} \right. \tag{7}$$

Moreover, the first equation of (7) has to be solved again in order to find the optimal step-length ρ of the gradient method at each optimization iteration α:

$$\gamma^{\alpha+1} = \gamma^{\alpha} - \rho j^{'}(\gamma^{\alpha})$$

2.3 Discrete problem

We consider the nozzle represented on Figure 2:

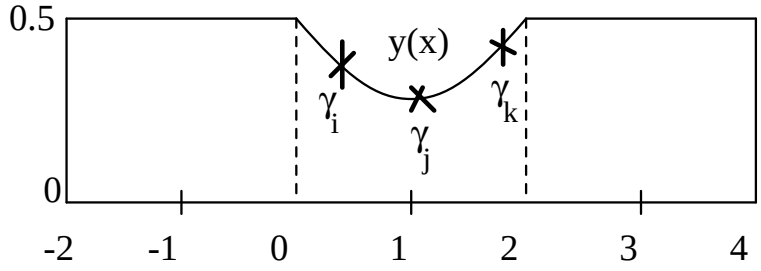


Figure 2: sketch of the computational domain

The part to optimize is for x in $[0, 2]$. The height, $y(x)$, is a variable curve. We have a triangular mesh where the abscissæare fixed and the ordinates change only for x in $[0, 2]$. We then use a "concertina" mesh (Figure 3).

We use a triangular upwind finite volume formulation, for each vertex i we have :

$$\Psi_{Euler}(W) = \sum_{j \in \kappa(i)} \Phi(W_{ij}, W_{ji}, \int_{\delta C_{ij}} \vec{n} \; d\sigma) \; + \; boundary \;\; conditions$$

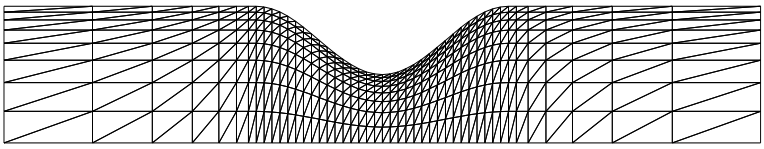


Figure 3: Concertina mesh with 423 nodes.

$\kappa(i)$ is the set of neighboring nodes of i, δC_{ij} is the intersection between cell's boundary of i and j, $\vec{n}$ is the outward normal vector on δC_i, Ψ is the Van Leer flux function differentiable. Therefore, to solve the discrete system for the Euler steady equations, we use an implicit linearized time advancing iteration :

$$(\frac{Id}{\Delta t} \; + \; \Psi'(W^n))\delta W^{n+1} \; = \; -\Psi(W^n) \tag{8}$$

where $\Psi'(W^n)$ is computed without approximation (we recall that when Δt is made arbitrarily large, this iteration approaches Newton's method).

The parameters of the optimization (or control parameters) are the ordinates of the nodes being on the part of the boundary to optimize : $\gamma = (\gamma_1, \ldots, \gamma_i, \ldots, \gamma_l)$. Let look at the sets on which j and J are defined.

$$\begin{array}{rccc} j : & I\!R^l & \longrightarrow & I\!R \\ & \gamma & \longmapsto & j(\gamma) \end{array}$$

$$\begin{array}{rccc} J : I\!R^l \times I\!R^{4m} & \longrightarrow & I\!R \\ (\gamma, W) & \longmapsto & J(\gamma, W) \end{array}$$

where l is the number of control parameters and m is the number of nodes of the mesh (we note that $l \ll 4m$, l is near $\sqrt{m}$).

3 One-Shot Method

In [2],[5],[6], is applied an optimization method which completely solves the linear systems (4) and (6). $W(\gamma)$ is solution of (4), the state equation and $\Pi(\gamma)$ is solution of (6), the adjoint-state equation. Both depend on γ. So, $\gamma = (\gamma_1, ..., \gamma_l)$ **is the only unknown** of the optimization problem (7). Meanwhile, although there is only one unknown, the optimization problem is very costly to solve, because of the solution of the two linear systems at each

step : by solving (4), we have to make many time-iterations (about seventy) in which we make sweeps Jacobi (sometimes a hundred). By solving (6), we still make a lot of sweeps Jacobi.

So, instead of solving completely each linear system, we solve simultaneously (7) (see a first approach in [12]). That means that $W(\gamma)$ becomes W, which is no more the solution of the state equation and $\Pi(\gamma)$ becomes Π, which is no more the solution of the adjoint-state equation. Then, after each iteration optimization, $j'(\gamma)$ which was the actual gradient in the complete solution of the systems, is now a number, that will only be the gradient at convergence. **We have now three unknowns** : W, Π and γ. We call this method the "one-shot" method, using so the denomination introduced by Ta'asan in [14].
In each step, we apply now one time-iteration for the flow Ψ, in which we only make three sweeps Jacobi and to compute the adjoint-state equation for Π , we make three sweeps Jacobi too !

3.1 Comparison between complete and one-shot solution methods

When the state equation for $\Psi(W(\gamma), \gamma)$ and the adjoint-state for $\Pi(\gamma)$ were solved completely, a descent method was applied because the actual gradient was calculated. Now, in the new approach, there is no more a descent method. So, it is no more an optimization problem because the cost functional may increase at some "optimization" iteration.

Complete solution method :

One solves :

$$\min_{\gamma \in I\!R^l} j(\gamma)$$

This means that we minimize j with respect to $\gamma = (\gamma_1 ... \gamma_l)$ (l shape-unknowns), with the constraint on the state equation : $\Psi(\gamma, W(\gamma)) = 0$.
The algorithm is :

$$\gamma^{\alpha+1} = \gamma^{\alpha} - \rho j'(\gamma^{\alpha}) \tag{9}$$

Lemma : With n unknowns, we have to minimize j at least n times.

However, before applying (9), we have to minimize j, which requires to compute the flow Ψ, with M unknowns for W ($M = 4 \times$ mesh nodes , $l \ll M$). We then have a complexity of $O(l \times M)$. So, if we double the mesh nodes, the complexity is of order of $O(2l \times M)$.

One-shot method :

The objective is to solve :

$$j'(\gamma, W, \Pi) = 0$$

in $\mathbb{R}^{l+M+M}$. Which means that we "minimize" j with respect to $\gamma = (\gamma_1 ... \gamma_l)$, W (with M unknowns) and Π (with M unknowns too).
There is a much larger number of unknowns because the 3 equations (state, adjoint-state and optimality condition) are assembled :

$$\begin{pmatrix} \text{state } (\gamma, W) \\ \text{adjoint-state } (\gamma, W, \Pi) \\ \text{optimality condition } (\gamma, W, \Pi) \end{pmatrix} = 0$$

Then, result of the previous Lemma can not be applied. The complexity becomes $O(l + M + M)$. If the mesh nodes are doubled, the complexity is not really larger. A mesh-independent convergence is then expected.

3.2 One-shot method

We solve a linear system of three equations

$$\begin{cases} \Psi(\gamma, W) = 0 & \text{(state)} \\ (\dfrac{\delta \Psi}{\delta W}(\gamma, W))^* \Pi = \dfrac{\delta J}{\delta W}(\gamma, W) & \text{(adjoint-state)} \\ j'(\gamma) = \dfrac{\delta J}{\delta \gamma}(\gamma, W) - < \Pi, \dfrac{\delta \Psi}{\delta \gamma}(\gamma, W) > & \text{(optimality condition)} \end{cases} \tag{10}$$

with 3 unknowns W, Π and γ.

We then evaluate control parameter with :

$$\gamma^{\alpha+1} = \gamma^{\alpha} - \rho j'(\gamma^{\alpha})$$

A sketch of the algorithm is presented in Figure 4.

Let explain more precisely what we really do to solve this system. We simplify (10) in:

$$\left\{ \begin{array}{rcll} E(\gamma, W) & = & 0 & \text{state} \\ A(\gamma, \Pi, W) & = & 0 & \text{adjoint-state} \\ G(\gamma, \Pi, W) & = & 0 & \text{optimality condition} \end{array} \right. \tag{11}$$

So we define Ω as :

$$\Omega = \begin{pmatrix} E(\gamma, W) \\ A(\gamma, \Pi, W) \\ G(\gamma, \Pi, W) \end{pmatrix}$$

and

$$\Omega' = \begin{pmatrix} \frac{\partial E}{\partial W} & 0 & \frac{\partial E}{\partial \gamma} \\ \frac{\partial A}{\partial W} & \frac{\partial E}{\partial W}^* & \frac{\partial A}{\partial \gamma} \\ \frac{\partial G}{\partial W} & \frac{\partial G}{\partial \Pi} & \frac{\partial G}{\partial \gamma} \end{pmatrix}$$

which is the Jacobian matrix of Ω. Its dimension is $(M + M + l)^2$ (M parameters for W and Π and l parameters for γ).

So, we want to solve :

$$\begin{pmatrix} W \\ \Pi \\ \gamma \end{pmatrix}^{\alpha+1} = \begin{pmatrix} W \\ \Pi \\ \gamma \end{pmatrix}^{\alpha} - \rho \, B^{-1} \begin{pmatrix} E(\gamma, W)^{\alpha} \\ A(\gamma, \Pi, W)^{\alpha} \\ G(\gamma, \Pi, W)^{\alpha} \end{pmatrix} \tag{12}$$

where B is near Ω', but it is a diagonal matrix, and then $B^{-1}y$ is not expensive to compute.

By looking at Ω', and more particularly at its two first lines, W (with γ and Π fixed) can be relaxed by a preconditioner B with zeros on the second and third columns. Π (with W and γ fixed) can also be relaxed by the same preconditioner B with zeros on the first and third columns. Then, the

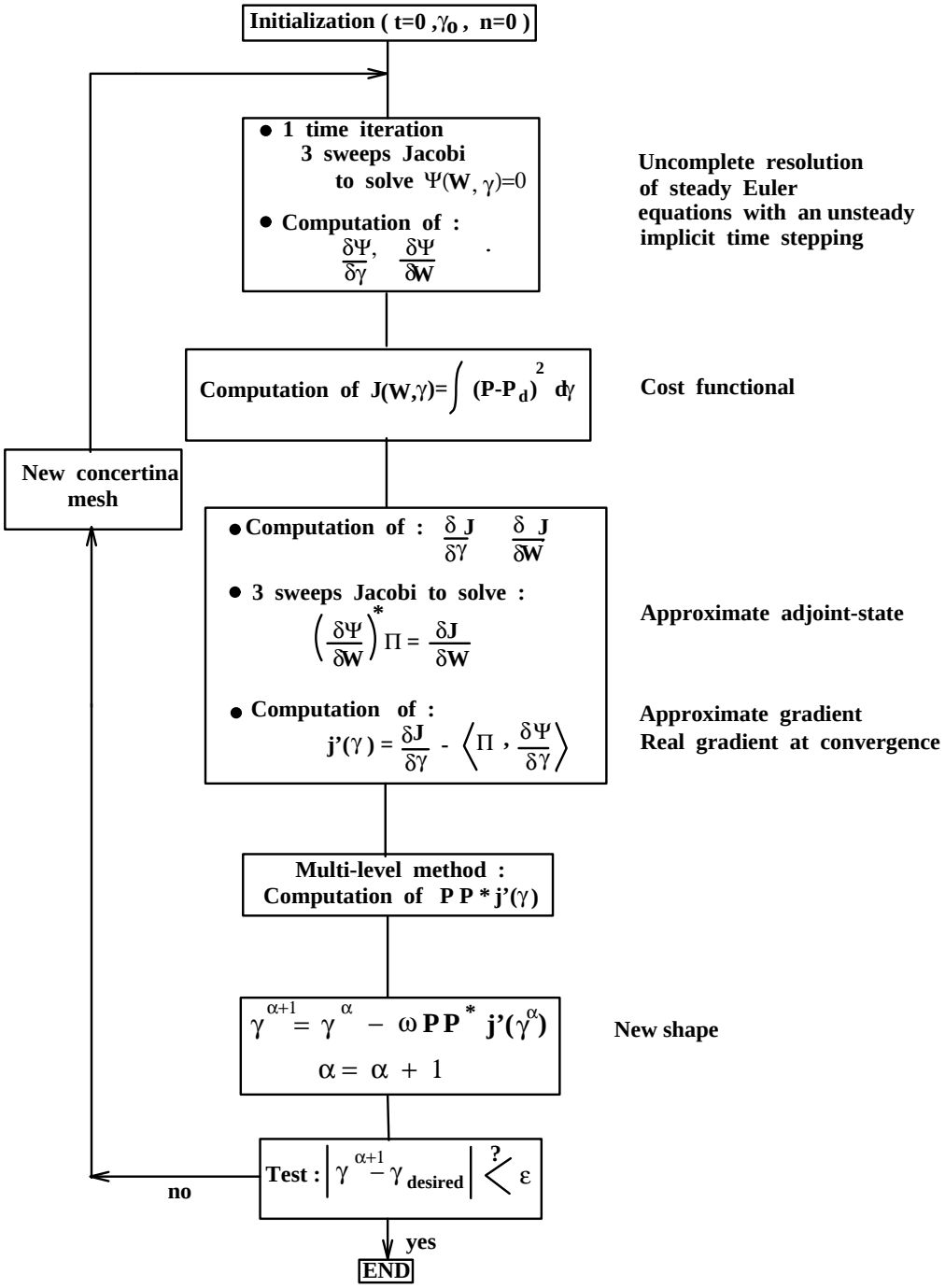


Figure 4: One-shot method.

partial derivatives $\frac{\partial E}{\partial W}$ and $\frac{\partial E}{\partial W}^*$ become diagonal matrices. $\frac{\partial E}{\partial W}$ and $\frac{\partial E}{\partial W}^*$ are M-matrices.

Ω' becomes :

$$\Omega' = \begin{pmatrix} \frac{\partial E}{\partial W} & 0 & 0 \\ 0 & \frac{\partial E}{\partial W}^* & 0 \\ \frac{\partial G}{\partial W} & \frac{\partial G}{\partial \Pi} & \frac{\partial G}{\partial \gamma} \end{pmatrix}$$

As regards the last line of the matrix, we know nothing about the terms $\frac{\partial G}{\partial W}$ and $\frac{\partial G}{\partial \Pi}$. Maybe we can omit them, maybe not. We then have omit them and put the Identity matrix instead of $\frac{\partial G}{\partial \gamma}$.
So, to solve the last unknown γ, the algorithm is :

$$\gamma^{\alpha+1} = \gamma^{\alpha} - \rho \; G(\gamma, W, \Pi)^{\alpha} \tag{13}$$

and then, we obtain a diagonal matrix :

$$\text{“}New\text{”}\Omega' = \begin{pmatrix} \frac{\partial E}{\partial W} & 0 & 0 \\ 0 & \frac{\partial E}{\partial W}^* & 0 \\ 0 & 0 & Id \end{pmatrix} = B$$

Meanwhile, this method to solve (12) is not a Jacobi method because the step ρ is not fixed from a level to another.

Research of an optimal constant step-length ρ :

The step ρ is constant on a given level. As we want to work with the most efficient one, some tests have been done before finding it. An efficient step is the bigger one that allows a convergence. For smaller steps, there is a slower convergence. Figure 5 presents the research of the step ρ for a complete solution with 7 parameters (we conclude that $\rho = 0.001$ is the optimal step). The same approach has been used for the simultaneous solution. We have conclude that $\rho = 6,5.10^{-4}$ was the optimal step for a level with 7 parameters; then for finer levels, we apply a length-step near twice smaller and for

coarser levels, we apply a length-step near twice bigger. The research of ρ is then not automated.

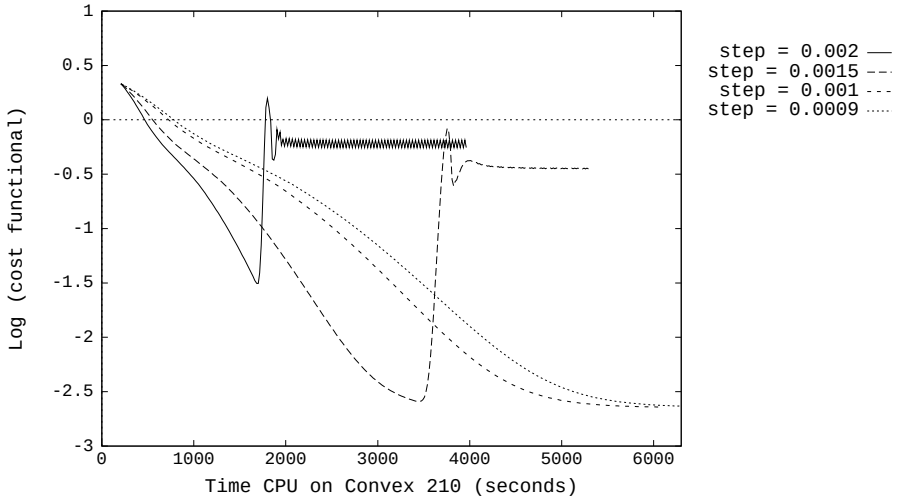


Figure 5: Research of the most efficient step. (7 parameters).

4 Numerical Results

Results are obtained on a mesh with 423 nodes and the multi-level parametrization cited in [3], [6] is applied. Calculations have been executed on a Convex 210.
The next paragraph exposes briefly multi-level method.

4.1 Multi-level method

The motivation was the fact of using all the parameters (the finest level) was too expensive and too slow. So, we have solved our problem alternately on the different levels (from coarsest to finest) because the problem was the same on each level and the fact of being on coarse levels implied a faster convergence.

- **Parametrization in optimization**

The optimization problem is :

$$\text{Find } \bar{\gamma} \text{ such that } j(\bar{\gamma}) = \min_{\gamma \in E} j(\gamma) \tag{14}$$

j is the cost functional (real valued functional) defined on an Hilbert E (fine level: N parameters, N components).
Let consider the problem of introducing a parametrization in an optimization process.
Let F be a second Hilbert space and P a linear continuous mapping from F to E (F is assimilated to the coarse level, n parameters, n components, $n \ll N$).
We have then the following iterative algorithm :

$$\begin{cases} \gamma_o & \text{given} \\ \gamma^{\alpha+1} = & \gamma^\alpha - \rho P P^* \Lambda j'(\gamma^\alpha) \end{cases} \tag{15}$$

This is a minimization in the subspace $P(F)$ (n parameters, N components) of E.
Λ is the canonic isomorphism between E' and E. $\Lambda j'(\gamma) = grad\ j(\gamma)$ is the gradient of j in E and $P^* grad\ j(\gamma)$ corresponds to the gradient in F.
Algorithm (15) is equivalent to apply the gradient method in F :

$$\text{Find } \bar{v} \text{ such that } j \circ P(\bar{v}) = \min_{v \in F} j \circ P(v) \tag{16}$$

Conclusion :
To minimize joP on F (coarse level) with a gradient method is equivalent to minimize j on $P(F)$ with a descent direction ($PP^* grad\ j$).

On the finest level the algorithm is :

$$\gamma^{\alpha+1} = \gamma^\alpha - \rho j'(\gamma^\alpha) \quad (PP^* = Id)$$

but on coarser levels it becomes :

$$\gamma^{\alpha+1} = \gamma^\alpha - \rho P P^* \Lambda j'(\gamma^\alpha)$$

PP^* is a $N \times N$ matrix but its rank is n.

- **Multi-level parametrization and interpolation**

  The number of parameters describes the coarseness of the parametrization or level. Coarser and finer levels will be applied alternately like in a multigrid process. P is an interpolation polynomial, see in [3]. Let $f \in F$. $(Pf)_j$ is directly defined from values $f_{i_1}, ..., f_{i_q}$ of f on existing coarse nodes : $(Pf)_j$ = linear combination of $f_{i_1}, ..., f_{i_q}$. (Pf) is defined from an implicit equation :

$$\sum_{k \in fine} \lambda_{i_k} (Pf)_k = \sum_{l \in coarse} \mu_{i_l} f_l$$

  In Figure 6, the symbol "$\bullet$" represents the values of f_i of the first parametrization, while the symbol "$*$", the interpolated values $(Pf)_j$; then, starting from the new set of points, we calculate by the same interpolation rule, an other set of points (symbolized in Figure 6 by "+"), and we repeat this procedure until all points are generated. This process yields smooth discrete curves (see [9], [10]) according to subdivide interpolation theory.

- **Multi-level optimization strategy**

  The basic idea is to calculate a descent direction with alternately different levels of parametrization. A V-cycle strategy is considered; it consists in the following phases :
  we start on the coarsest level and we make an optimization iteration on all next finer levels. When we are on the finest level, we start again; we make an optimization iteration on all next coarser levels. We have done a V-cycle. The strategy is plotted on Figure 7 for 3 levels.

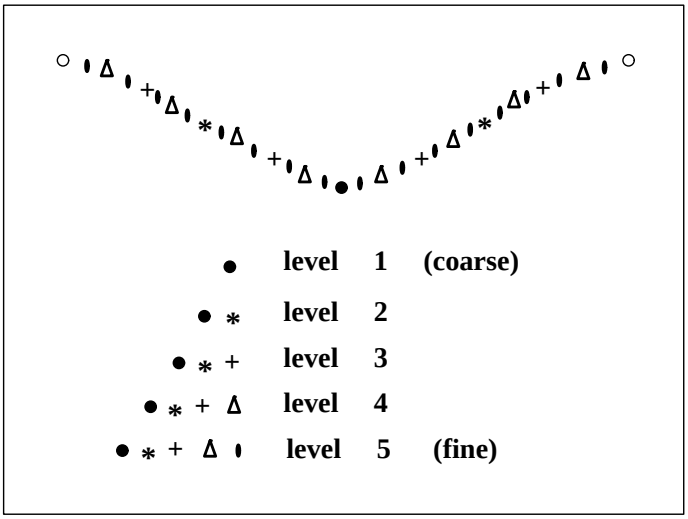


Figure 6: Sketch of the parametrization.

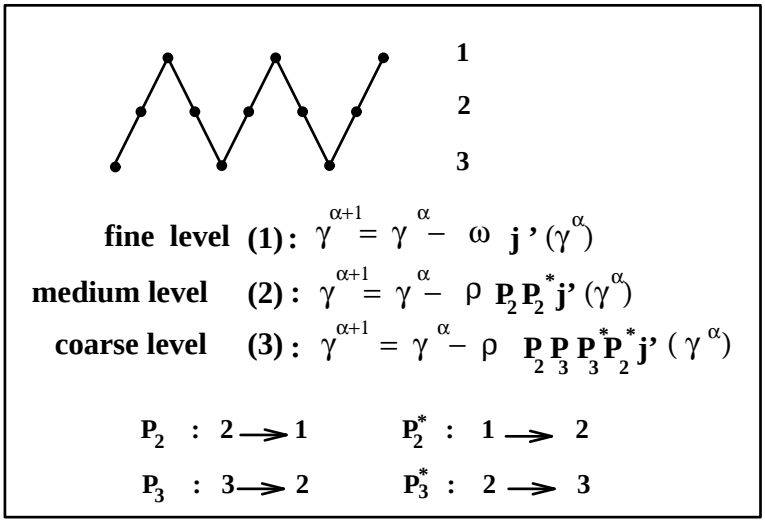


Figure 7: V-cycle strategy for 3 levels.

4.2 Numerical results

At first, we have calculated sweeps Jacobi needed to obtain a flow (or a cost functional). The implicit linearized time advancing iteration (8) has been solved with a remainder $||\Psi|| = 10^{-6}$: 300 sweeps Jacobi were needed for one cost functional evaluation. So, one output will be to compare the number of sweeps Jacobi in complete solution and in one-shot method.

It is true that in the one-shot method we do not calculate the cost functional at each "optimization" iteration, but we use sweeps Jacobi. So, we can have an estimation about the number of cost functionals calculated in such a method. In Figures (8), (9), (10), and (13) we have in abscissæthe cost functionals evaluation which is in fact sweeps Jacobi divided by 300.

Note that the evaluation of a cost functional demands 50 "optimization" iterations and 115 seconds CPU (on a CONVEX 210), for the one-shot method.

In Figure 8, we have considered 15 and 7 control parameters, and we have analysed the cost functional convergence first with the complete solution and after with simultaneous solution (for 1 optimization iteration : 1 time-iteration and 3 sweeps jacobi for the state equation and 3 sweeps jacobi for the adjoint-state equation). For 15 parameters, we have taken for the optimal step, $\rho = 9.10^{-4}$ for the complete solution and $\rho = 3.10^{-4}$ for the one-shot method. For 7 parameters, we have taken for the optimal step, $\rho = 6,5.10^{-4}$ for the one-shot method and $\rho = 10^{-3}$ for the complete solution (as there is a little number of parameters, cost functional decreases less lower).

As regards 7 parameters, we have converged to 10^{-3} with a gain of 54.5 for the cost functionals evaluation and 13.5 for time CPU. With 15 control parameters, we have an estimation of the number of cost functionals: for complete solution, at convergence, more than 1200 are computed for a time CPU larger than 40000 seconds ! With one-shot method we obtain a gain of about 35 for cost functioanl and 10 for time CPU. We then note that one-shot is efficient.

By using one-shot combined with **multi-level parametrization**, we have alternated first with 7 and 15 parameters and after, with $3, 7$ and 15 parameters, and have compared with one-shot on the 15-parameters level (see Figure 9).

When we use multi-level by alternating on two levels (7/15), convergence as regards cost functionals evaluation or time CPU is 2.33 times faster than convergence on the 15 parameters level. If we alternate with three levels (3/7/15), convergence is still faster with a factor 3.4 . We have converged in ten steps for fifteen shape unknowns !

Figure 10 compares complete solution and one-shot method both combined with multi-level parametrization. In complete solution, the strategy was used with all the parameters (3/7/15/31), see [3], and it was with a Full V-cycle method. For one-shot method, we have used V-cycles alternating on the 3 previous levels. There is a gain of 35 as regards the cost functionals evaluation !

In Table 1, we can see the difference between complete solution and one-shot method combined with multi-level parametrization.

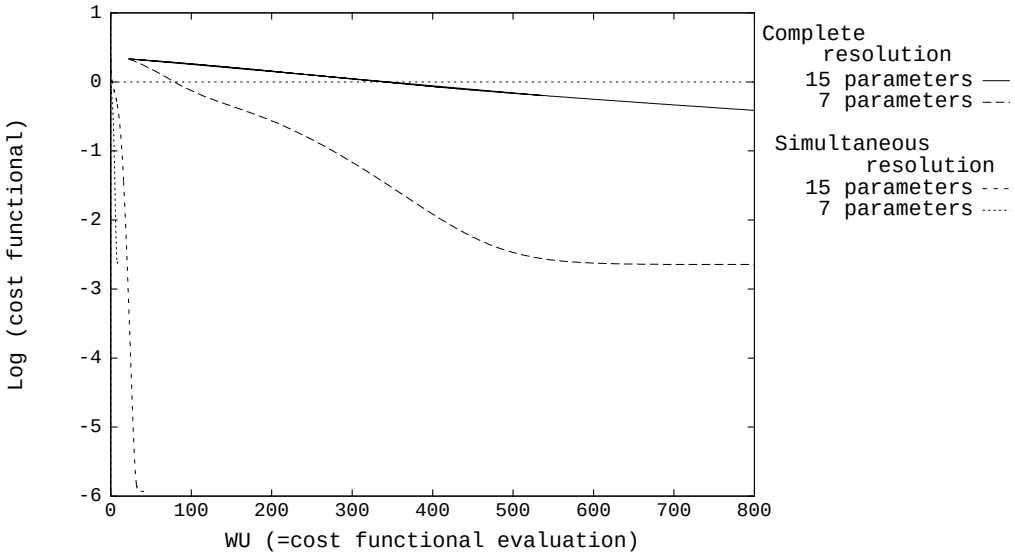


Figure 8: Comparison between one level One-Shot and Complete Resolution (7 and 15 parameters).

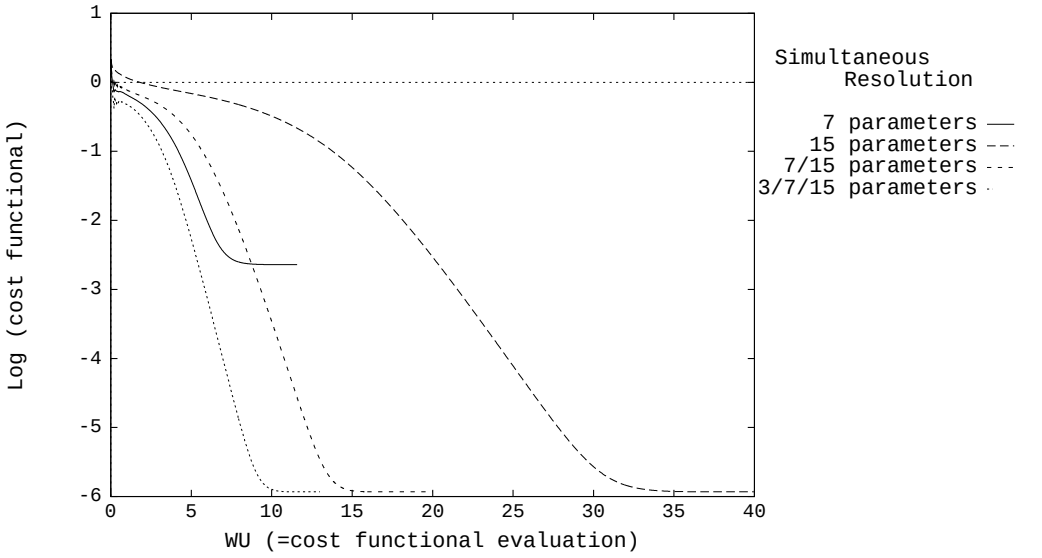


Figure 9: One-shot combined with multi-level method.

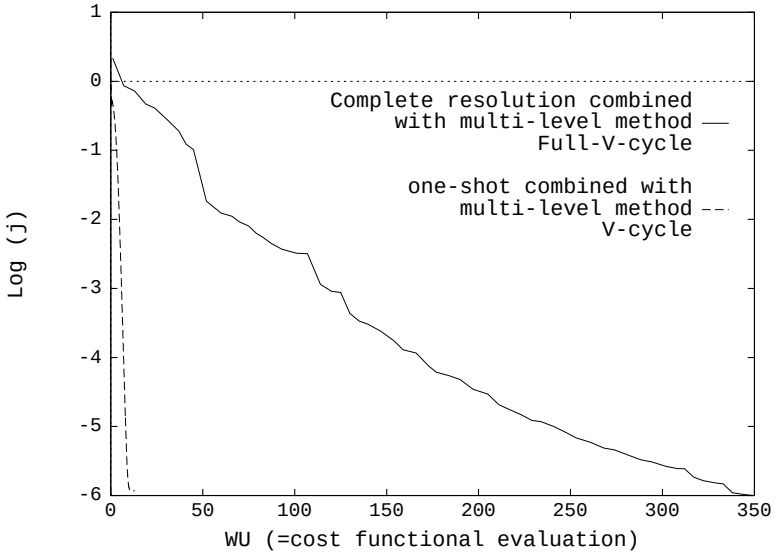


Figure 10: Complete [3] and uncomplete solutions combined with multi-level method.

	7 parameters		**15 parameters**	
	cost functional evaluation	time CPU (seconds)	cost functional evaluation	time CPU (seconds)
COMPLETE RESOLUTION	log (j)= -2.64		log (j) = -5.93	
	6 0 0	1 9 0 0 0	> 1 2 0 0 **(estimated)**	> 4 0 0 0 0 **(estimated)**
ONE - SHOT ON ONE LEVEL	log (j)= -2.64		log (j) = -5.93	
	1 1 **(equivalent)**	1 4 0 0	3 6 **(equivalent)**	4 4 0 0

	7-15 parameters		**3-7-15 parameters**	
	cost functional evaluation	time CPU (seconds)	cost functional evaluation	time CPU (seconds)
ONE - SHOT COMBINED WITH MULTI - LEVEL	log (j) = -5.93		log (j) = -5.93	
	1 5 **(equivalent)**	1 9 0 0	1 0 **(equivalent)**	1 3 0 0

Table 1: Evaluations of cost functionals for complete solution and one-shot method combined with multi-level method.

We can see, in Figure 11, the successive shapes for the one-shot method combined with multi-level parametrization with 3/7/15 parameters. There is convergence to the desired shape after 400 "optimization" iterations (1000 sec. CPU).

Figure 12-a, shows two-grids-ideal method (where we have alternated with 7 and 15 parameters) which consists in making one optimization iteration on the fine level and solve completely the problem on the coarse level. We still use V-cycles. We see on Figure 12-b that the method becomes "ideal" when we relax the coarse level with at least 450 iterations. It implies that at the

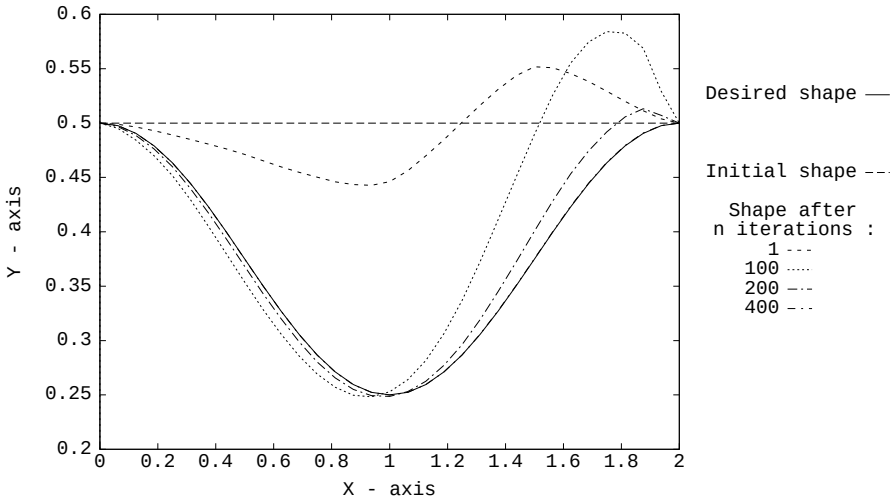


Figure 11: Successive shapes with One-Shot combined with multi-level parametrization (3/7/15).

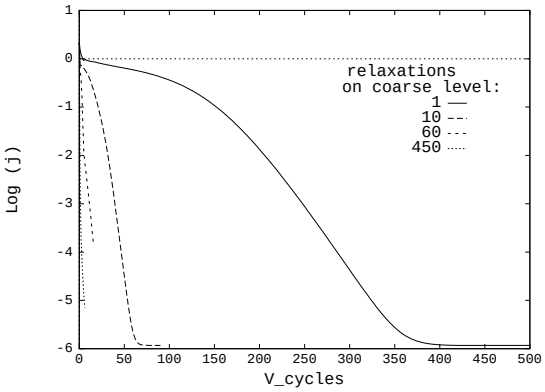
first V-cycle, we have completely solved on the coarse level ($log(j) = -2.64$). As everyone knows it, this method is very expensive (the number of the evaluations of cost functionals is about 54 !), but it shows that the dumping of the error is mesh-independent. Unfortunately, we did not succeed in obtaining a 2-Grids-Ideal method by alternating with 15 and 31 parameters, because of the step-length ρ on the finest level. In fact, we did not manage to find the optimal one.

Let us examine the ideal arithmetic complexity of the two methods (complete and simultaneous solutions) in the case of optimal order solution algorithms.

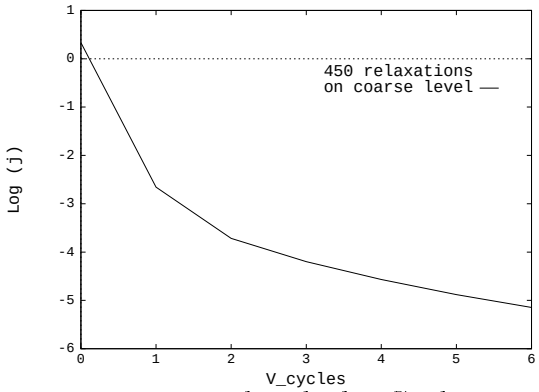
Complete solution combined wih multi-level method:

The cost is $O(k(\log k)^2)$ where $k = l_\gamma$ is the number of shape parameters of the optimization ($\gamma = (\gamma_1, .., \gamma_{l_\gamma})$).
So, if we double this number of parameters, the cost becomes :

$$O(\ (2l_\gamma(\log l_\gamma)^2)(\epsilon_{l_\gamma} + 1)^2\ )$$



a- Research of Ideal-2-Grids.



b- Ideal-2-Grids.

Figure 12: Ideal-2-Grids with 7 and 15 parameters.

where $\epsilon_{l_\gamma} = \dfrac{\log 2}{\log l_\gamma}$.
Then, the cost is more than twice greater than $O(k(\log k)^2)$!

Simultaneous solution combined wih multi-level parametrization:

The cost is $O(k(\log k)^2)$ where $k = l_\gamma + M_\Pi + M_W$ where $M_\Pi = M_W = 4 \times$ mesh nodes.
So, if we double l_γ, the complexity $O(\ (2l_\gamma + M_\Pi + M_W) log^2(2l_\gamma + M_\Pi + M_W)\ )$ is nearly the same than previous one, because $2l_\gamma \ll M_\Pi + M_W$

So, we can hope a cost which is quasi-independant of the number of the parameters.

Concluding remarks

In a first approach, we have seen that by solving simultaneously the flow equation and shape optimality, we have reduced the calculation time of a factor equal to 13.5 for 7 parameters and about 10 for 15 parameters. As regards cost functional evaluations, we have a gain of 54.5 for 7 parameters and 80 for 15 parameters. One-shot method is thus efficient !

In a second approach, we have combined one-shot method with multi-level parametrization. By alternating on 3 levels (3/7/15), results are still

more efficient. In comparison with one-shot on the 15 parameters level, there is a gain of 3.6 for cost functional evaluation and 3.4 for time CPU.

However, we have tried to alternate with all the parameters (3/7/15/31), but results are not yet very convincing. One possible difficulty is the search of the optimal relaxation step-length ρ on the finest grid (31 parameters); in fact, we did not succeed in finding the optimal one. The chosen step seems to be very small. Then, the fact of adding the 31 parameters level slows down the convergence (see Figure 13).

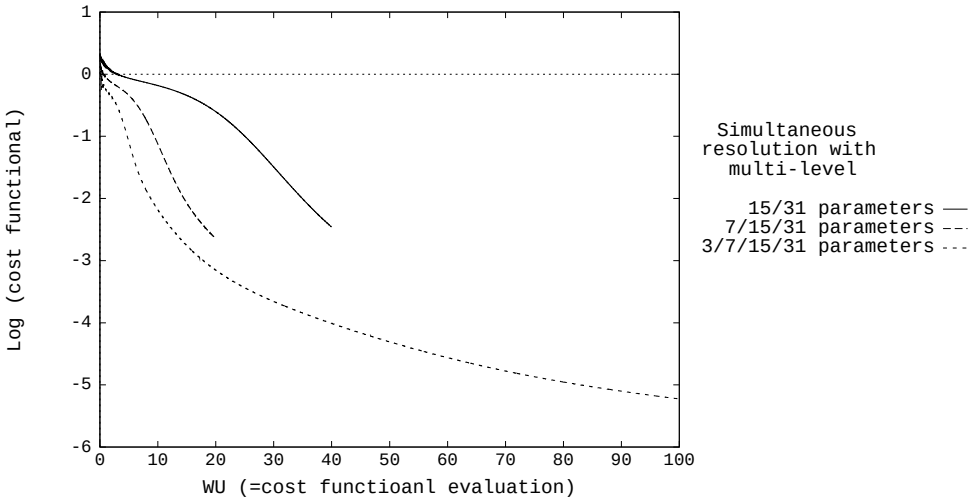


Figure 13: One-shot combined with multi-level method with all the parameters.

The search of the optimal step of optimization is not automated. So, before having these good results, we must test some values. Then, it may be necessary, later on, to consider a one-shot method where the optimal step is calculated at each optimisation iteration, as in [14] (we can see a first investigation in this direction in [12]).

Acknowledgements

We thank A. Dervieux for having advised and directed us in our investigations.

F. Beux has been partly supported by Brite Euram 1082 Contract at INRIA - Sophia Antipolis and by MRE fellowship at Pisa.

References

[1] F. ANGRAND, R. GLOWINSKI, J. PERIAUX, P. PERRIER, O. PIRONNEAU, G. POIRIER *"Optimum Design for Potential Flow"*, Proc. Finite Elements in Flows Problems. Calgary (1980).

[2] F. BEUX, A. DERVIEUX *"Exact-gradient shape optimization of a 2D Euler flow"*, Finite Elements in Analysis and Design, Vol. **12**, pp. 281-302 (1992).

[3] F. BEUX, A. DERVIEUX *"A hierarchical approach for shape optimization"*, to appear in Engineering Computations, INRIA Research Report No 1868 (1993).

[4] F. BEUX *"Un algorithme d'optimisation d'une forme aérodynamique par une méthode de gradient exact"*, Proc. of the Colloque "Optimisation and Control" organised for the sixtieth birthday of J. Cea, editors : J.A Desideri, L. Fezoui, B. Larrouturou and B. Rousselet, editions CÉPADUÈS, Toulouse, pp. 73-88 (1993).

[5] F. BEUX *"Shape optimization of an Euler flow in a nozzle"*, Proceedings of BRITE EURAM 5 "Optimum Design In Aerodynamics", J. Périaux editor, to appear in "Notes on Numerical Fluid Mechanics", editions Vieweg Verlag, Braunsweig-Wiesbaden (1993).

[6] F. BEUX, N. MARCO, H. GUILLARD, A. DERVIEUX *"Multilevel Optimization : Application to P.D.E Solution and to Shape Optimum Design"*, Proc. of "VIII International Conference on Finite Elements in Fluids", Barcelona, (1993).

[7] M. BORSI *"Aerodynamic design and optimisation at Alenia D.V.D"*, Computational Methods in Applied Sciences, ACCOMAS, Brussels (1992).

[8] M. O. BRISTEAU *"Application of a Finite Element Method to Transonic Flow Problems using an Optimal Control Approach"*, Von Karman Institute for Fluid Dynamics Rhodes-Saint-Genèse (Belgium), Lecture Series 1978 - 4.

[9] G. DESLAURIERS, S. DUBUC *"Symmetric iterative interpolation schemes"*, Constructive Approximation, Vol. 5, pp. 49-68 (1989).

[10] N. DYN, J.A. GREGORY, D. LEVIN *"Analysis of linear binary subdivision schemes for curve design"*, Constructive Approximation, Vol. 7, pp. 127-147 (1991).

[11] A. JAMESON *"Aerodynamic Design via Control Theory"*, Report 1824 MAE, Princeton University (1988).

[12] N. MARCO *"Algorithmes d'Optimisation d'une forme aérodynamique"*, Report of DEA "Turbulence et Systèmes Dynamiques", Nice University (1992).

[13] V. SELMIN, M. BORSI *"Direct optimization of transonic airfoils"*, Brite-Euram "Optimum Design in aerodynamics" contract AERO-0026C proposal 1082, MidTerm report (1991).

[14] S. TA'ASAN *"One Shot Methods for Optimal Control of Distributed Parameter Systems. I : Finite Dimensionnal Control"*, ICASE Report No 91-2, NASA Contractor Report 187497 (1991).

[15] S. TA'ASAN, G. KURUVILA *"Aerodynamic Design and Optimization in One-Shot"*, AIAA paper 92-0025 (1992).

Unité de recherche INRIA Lorraine, Technôpole de Nancy-Brabois, Campus scientifique,
615 rue de Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399